CORe50: a New Dataset and Benchmark for Continuous Object Recognition

Vincenzo Lomonaco & Davide Maltoni
Department of Computer Science and Engineering - DISI
Alma Mater Studiorum - University of Bologna
http://vlomonaco.github.io/core50
{vincenzo.lomonaco, davide.maltoni}@unibo.it

Abstract

Continuous/Lifelong learning of high-dimensional data streams is a challenging research problem. In fact, fully retraining models each time new data become available is infeasible, due to computational and storage issues, while naïve incremental strategies have been shown to suffer from catastrophic forgetting. In the context of real-world object recognition applications (e.g., robotic vision), where continuous learning is crucial, very few datasets and benchmarks are available to evaluate and compare emerging techniques. In this work we propose a new dataset and benchmark *CORe50*, specifically designed for continuous object recognition, and introduce baseline approaches for different continuous learning scenarios.

1 Introduction

Datasets such as *ImageNet* and *Pascal VOC* provide a very good playground for classification and detection approaches. However, they have been designed with "static" evaluation protocols in mind; the entire dataset is split in just two parts: a training set is used for (one-shot) learning and a separate test set is used for accuracy evaluation. Splitting the training set into a number of batches is essential to train and test continuous learning approaches, a hot research topic that is currently receiving much attention. Unfortunately, most of the existing datasets are not well suited to this purpose because they lack a fundamental ingredient: the presence of multiple (unconstrained) views of the same objects taken in different sessions (varying background, lighting, pose, occlusions, etc.). Focusing on Object Recognition we consider three continuous learning scenarios:

- **New Instances** (**NI**): new training patterns of the same classes become available in subsequent batches with new poses and conditions (illumination, background, occlusion, etc.). A good model is expected to incrementally consolidate its knowledge about the known classes without compromising what it has learned before.
- **New Classes** (**NC**): new training patterns belonging to different classes become available in subsequent batches. In this case the model should be able to deal with the new classes without losing accuracy on the previous ones.
- **New Instances and Classes** (**NIC**): new training patterns belonging both to known and new classes become available in subsequent training batches. A good model is expected to consolidate its knowledge about the known classes and to learn the new ones.

It appears clear that for dealing with such complex scenarios datasets/benchmarks specifically designed for continuous learning are needed in order to evaluate and compare emerging approaches. In particular, the ideal dataset should be composed of a high number of classes to be added in subsequent batches, but, more important, of a large number of views for each class acquired in different sessions. The presence of temporal coherent sessions (i.e., videos where the objects gently

move in front of the camera) is another key feature since temporal smoothness can be used to simplify object detection, improve classification accuracy and to address unsupervised scenarios [16] [17]. In Section 2 we review existing datasets and better clarify the motivations that induced us to propose a new benchmark. When addressing real-world continuous learning, assuming that the whole past data can be used at each training step (i.e., cumulative approach) is not only very far from biological learning but also unlikely for application engineering. In fact, the cumulative approach would require:

- To store all the previous data streams;
- To retrain a model on all the whole data each time new data is available.

Updating an already trained model with new data (only) is much more feasible in term of computation and memory constraints. Recent advances in transfer learning/tuning with deep neural networks have shown that using previously learned knowledge on similar tasks can be useful for solving new ones [15]. Yet, little has been done in the context of continuous learning where the same model is required to solve new tasks while maintaining good performances on the previous ones. Indeed, preserving previously learned knowledge without re-accessing old patterns remains particularly challenging due to the phenomenon known in literature as catastrophic forgetting where what has been learned so far is seriously compromised. The contributions of this paper can be summarized as follows:

- Collection of a new dataset (called *CORe50*) specifically designed for continuous object recognition. *CORe50* is publicly available at `http://vlomonaco.github.io/core50`.
- Definition of benchmarks for NI, NC, NIC scenarios on *CORe50*.
- Design and test of simple continual learning approaches for NI, NC, NIC scenarios to be used as baseline references when developing new approaches.

In section 2 we review related literature and compare datasets that can be used for continuous object recognition. Section 3 describes *CORe50*. While the new dataset is intended for continuous object recognition, in Section 4 we provide an overview of the accuracy that can be achieved by training recent CNNs on the whole training data (i.e., static benchmark). In Section 5 we introduce, for each continuous learning scenario (NI, NC and NIC), one or more simple approaches that can be used as baseline references when testing new developments on *CORe50*. Finally, in Section 6 some conclusions are drawn.

2 Related Works

In the last few years, we have witnessed a renewed interest in continuous learning, both for supervised classification and reinforcement learning. Several interesting approaches have been proposed such as: Learning without Forgetting [13], Progressive Neural Networks [23], Active Long Term Memory Networks [7], Adaptive Convolutional Neural Network [29], PathNet [5], Incremental Regularized Least Squares [3], Elastic Weight Consolidation [11], Encoder-based Lifelong Learning [28], etc.

Elastic Weight Consolidation (*EWC*) appears particularly intriguing since it provides a formal criterion to identify the subset of weights that should not be changed (or changed as little as possible) during the model update to reduce forgetting. *Learning without Forgetting* (*LwF*) [13] and some of its evolutions [28] are also very interesting since they prove that enforcing the output stability helps to control forgetting and, at the same time, provides enough degrees of freedom to learn the new task(s). Unfortunately, until now, most of the experimentations have been performed in setups that are appropriate to learn very short sequences of big tasks (typically 2), while their capabilities have not been assessed in continuous learning scenarios characterized by frequent (sometimes small) updates. For example, *EWC* classification tests have been carried out on permuted *MNIST* task [9] [26], where the new tasks to learn are obtained by scrambling the pixel positions in the *MNIST* dataset. Although original and ingenious such experiment is not truly linked to real-world applications. Most of the experiments performed on *LwF* consider pairs of large datasets (selected from *ImageNet*, *Pascal VOC*, *Places365-standard*, *Caltech-UCSD Birds-200-2011*, *MIT indoor scene classification*, etc) and proves that a model trained on the first dataset can be further trained on the second one without forgetting the first task; even if *LwF* can deal with multiple new tasks, only a simple experiments is reported in [13] for multiple new tasks. Furthermore, only the NC scenario has been addressed.

In Table 1 we compare datasets/benchmarks which, in our opinion, could be used for continuous object recognition. Datasets where temporal coherent sequences are not available (or cannot be generated

Table 1: Comparison of datasets (with temporal coherent sessions) for continuous object recognition.

Dataset	Cat.	Obj.	Sess.	Frames per sess.	Format	Acquisition setting	Outdoor sessions
NORB [12]	5	25	20	20*	grayscale	turntable	no
COIL-100 [18]	-	100	20	54*	RGB	turntable	no
iLab-20M [2]	15	704	-	-	RGB	turntable	no
RGB-D [24]	51	300	-	-	RGB-D	turntable	no
BigBIRD [25]	-	100	-	-	RGB-D	turntable	no
ALOI [8]	-	1000	-	-	RGB	turntable	no
BigBrother [6]	-	7	54	$\sim$20	RGB	wall cameras	no
iCubWorld28 [19]	7	28	4	$\sim$150	RGB	hand hold	no
iCubWorld-Transf [20]	15	150	6	$\sim$150	RGB	hand hold	no
CORe50	**10**	**50**	**11**	**$\sim$300**	**RGB-D**	**hand hold**	**yes (3)**

* Temporal coherent training/test sessions for NORB and COIL-100 have been defined in [16] and [17].

from static frames) are here excluded. In principle, such datasets could be used for continuous learning as well, but we think that temporally coherent sequences allow a larger number of real-world applications to be addressed (e.g., robotic vision scenario). *YouTube-8M* [1] provides a huge number of videos acquired in difficult natural settings. However, the classes are quite heterogeneous and acquisition conditions are completely uncontrolled in terms of object distance, pose, lighting, occlusions, etc. In other words, we believe it is too challenging for current continuous learning approaches (still in their infancy).

In the first group of datasets in Table 1 (*NORB*, *COIL-100*, *iLAB20M*, *Washington RGB-D*, *BigBIRD*, *ALOI*), objects are positioned on turntables and acquisition is systematically controlled in term of pose/lighting. Neither complex backgrounds nor occlusions are present in these datasets. For *NORB* and *COIL-100* we defined in [16] a number of exploration sequences that turn the native static benchmarks into continuous learning tasks; [16] also reports supervised and semi-supervised accuracy for the NI scenario. Exploration sequences can be generated for the other datasets in this group as well by randomly walking through adjacent static frames in the multivariate parameter space; however, the obtained sequences would remain quite unnatural. *BigBrother* dataset (see [15] [6]) is an interesting incremental learning setup in the face recognition domain, but unfortunately the owner copyright does not allow the public diffusion of the dataset. Finally, the *iCubWorld* datasets ([19], [20]) have been acquired in a robotic vision context and are the closest ones to *CORe50*. In fact, objects are hand hold at nearly constant distance from the camera and are randomly moved. With respect to these datasets, *CORe50* consists of a higher number of longer sessions (including outdoor ones), more complex backgrounds and also provide depth information (that can be used as extra-feature for classification and/or to simplify object detection).

In our opinion, the most important feature of *CORe50*, is the presence of 11 distinct acquisition sessions per object; this allows to define incremental strategies that are long enough to appreciate the learning trends. While preparing this paper we noted that *iCubWorld-Transf* is being expanded (see `https://robotology.github.io/iCubWorld/` for latest updates), and we think that cross-evaluating continuous object recognition approaches on both *CORe50* and *iCubWorld-Transf* could be very interesting.

3 CORe50

CORe50, specifically designed for (C)ontinuous (O)bject (Re)cognition, is a collection of 50 domestic objects belonging to 10 categories: plug adapters, mobile phones, scissors, light bulbs, cans, glasses, balls, markers, cups and remote controls (see Figure 1). Classification can be performed at object level (50 classes) or at category level (10 classes). The first task (the default one) is much more challenging because objects of the same category are very difficult to be distinguished under certain poses. The dataset has been collected in 11 distinct sessions (8 indoor and 3 outdoor) characterized by different backgrounds and lighting. For each session and for each object, a 15 seconds video (at 20 fps) has been recorded with a Kinect 2.0 sensor [27] delivering 300 RGB-D frames. Objects are

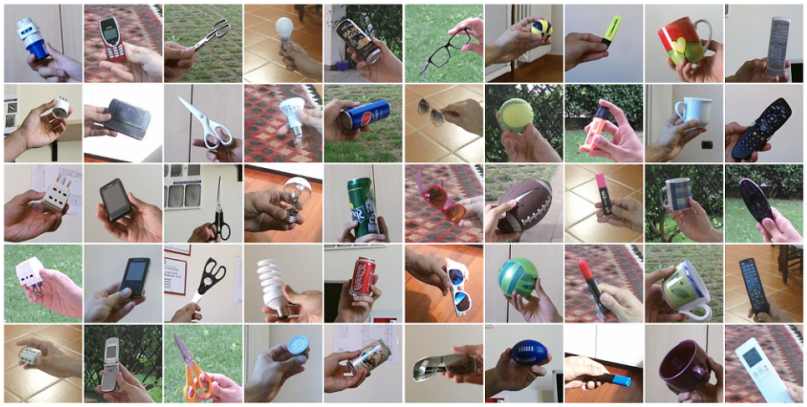

Figure 1: Example images of the 50 objects in *CORe50*. Each column denotes one of the 10 categories.

hand hold by the operator and the camera point-of-view is that of the operator eyes. The operator is required to extend his arm and smoothly move/rotate the object in front of the camera. A subjective point-of-view with objects at grab-distance is well-suited for a number of robotic applications. The grabbing hand (left or right) changes throughout the sessions and relevant object occlusions are often produced by the hand itself.

Row data consists of 1024×575 RGB + 512×424 Depth frames. Depth information can be mapped to RGB coordinates upon calibration. The acquisition interface identifies a central region where the object should be kept (see red box in Figure 2). This allows to performs a first (fixed) cropping, thus reducing the frame size to 350×350.

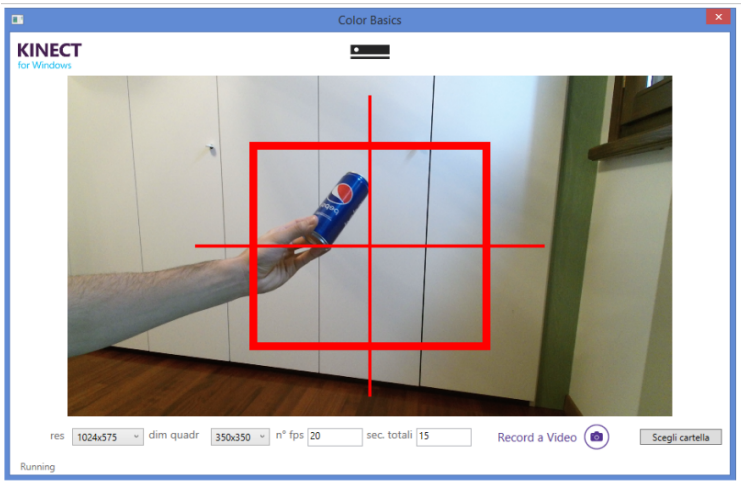

Figure 2: Acquisition interface: the red box identifies the central region where the operator is required to keep the objects while moving and rotating them.

Since our domestic objects (kept at arm distance) typically extend for less than 100×100 pixels, only a small fraction of the frame contains the object of interest. Therefore, we exploited temporal information to crop from each 350×350 frame a 128×128 box around the object[1]. To this purpose we implemented a simple but effective motion-based tracker working only on RGB data, so that a similar approach could be used even if depth information is not available (see Figure 3 for an

[1] in principle object detection inside the frame could be performed by a trained model like a Faster R-CNN [22] or Yolo [21]; however, since our main interest is continuous object recognition, at this stage we preferred to focus on object classification instead of the more complex and time demanding detection task.

example). While in most of the cases the objects are fully contained in the crop window, sometimes they can extend beyond borders (e.g., this can happen if the object distance from the camera is reduced too much, or the tracker partially loses the object because of a too fast movement). No manual correction has been applied, because we believe that tracking imperfections are unavoidable and should be properly dealt with at later processing stages.

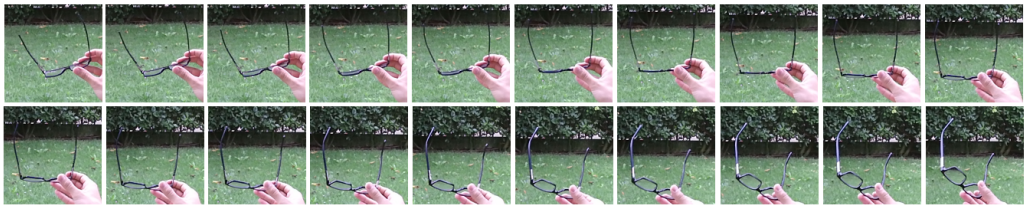

Figure 3: Example of 1 second recording (at 20 fps) of object #26 in session #4 (outdoor). Note the smooth movement, pose change and partial occlusion. The 128 $\times$ 128 frames here shown have been automatically cropped from 350 $\times$ 350 images based on a fully automated tracker.

The final dataset consists of 164,866 128 $\times$ 128 RGB-D images: 11 sessions x 50 objects x ($\sim 300^2$) frames per session. Figure 3 shows one frame of the same object throughout the eleven sessions. Three of the eleven sessions (#3, #7 and #10) have been selected for test and the remaining 8 sessions are used for training. We tried to balance as much as possible the difficulty of training and test sessions with respect to: indoor/outdoor, holding hand (left or right) and complexity of the background.

Figure 4: One frame of the same object (#41) throughout the 11 acquisition sessions. Note the variability in terms of background, illumination, blurring, occlusion, pose and scale.

The full dataset, along with further information can be downloaded from `vlomonaco.github.io/core50`. In the same repository we make available the code for the reproducibility of the benchmarks described in the following sections.

4 Static Object Recognition Benchmark

While designed for continuous learning, *CORe50* dataset can still be used as a medium size benchmark for object recognition with a static evaluation protocol. The high object pose variability and complex acquisition setting make the problem sufficiently hard to solve even when learning is performed on the whole training data.

In Table 2, we show the accuracy of two well-known CNN models (CaffeNet and VGG[3]) adapted to medium size and trained in three different modalities by using RGB data only (depth information will be used in future studies):

1. Mid-CNN from scratch: training a model from scratch.

[2]Some sequences are slightly shorter than 300 frames because a few initial frames are necessary to initialize the automatic motion-based tracker.

[3]We refer to the VGG-CNN-M model introduced in [4].

2. Mid-CNN + SVM: using a model pre-trained on *ILSVRC-2012* as a fixed feature extractor in conjunction with a linear SVM classifiers. Features are extracted at *pool5* level.
3. Mid-CNN + FT: fine-tuning an *ILSVRC-2012* pre-trained model on *CORe50*.

As already shown by many authors, fine-tuning a pre-trained model on the new dataset is often the most effective strategy, especially if the new dataset is large enough to avoid overfitting, but not so large to learn representative features from scratch.

Table 2: Accuracy of CaffeNet and VGG models (both adapted to size 128 × 128) on *CORe50* for different learning strategies. The test set consists of sessions: #3, #7 and #10; the training set of the remaining 8 sessions.

	Accuracy % **(object level: 50 classes)**		**Accuracy %** **(category level: 10 classes)**	
Strategy	**CaffeNet**	**VGG**	**CaffeNet**	**VGG**
Mid-CNN from scratch	37,82%	38,09%	48,93%	53,74%
Mid-CNN + SVM	51,35%	59,03%	61,81%	68,94%
Mid-CNN + FT	65,98%	69,08%	77,76%	80,23%

The term Mid-CNN is here used to highlight that we are not using the original 227 × 227 CaffeNet and 224 × 224 VGG models but their adaption to a mid-size of 128 × 128 pixels. Many researchers use available pre-trained CNN models as they are, and simply stretch their images to fit the model input size, even if the image size is much smaller than the CNN input. Stretching our input pattern (from 128 × 128 to 227 × 227) would require much more computation at inference time (about four times), so we decided to adapt the pre-trained CNN models to work with 128 × 128 input images. However, in case of pre-trained models, this step is not neutral and obvious as one could expect: more details are provided in Appendix A.

To improve classification accuracy, instead of classifying single frames, a set of temporally adjacent frames can be fused. To this purpose we implemented a simple sum-rule fusion at confidence level. The graph in Figure 5 shows the result for the Mid-VGG model. For each classification experiment (object level and category level) we tested two cases: *i*) we concatenate frames from all test sequences without considering end-of-sequence events (reset); *ii*) we assume that a reset signal is available. In the former, as the window size increases the risk of fusing frames from different classes increases as well. In general, fusing 40-50 frames (about 2 seconds of video) seems to be a good compromise even when sequences cannot be reliably segmented.

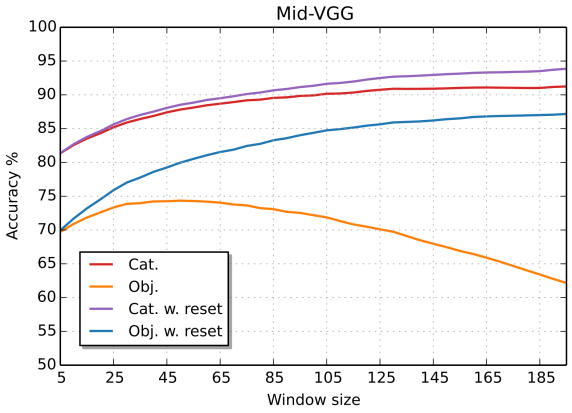

Figure 5: Mid-VGG classification accuracy (at object level and category level) when classification confidence over more adjacent frames is fused. On the horizontal axis the number of frames fused (temporal window). When end-of-sequence reset is not available using long temporal windows can lead to dangerous drifts (see the orange curve).

5 Continuous Object Recognition Benchmark

For all the continuous learning scenarios (NI, NC, NIC) we use the same test set composed of sessions #3, #7 and #10. The remaining 8 sessions are split in batches and provided sequentially during training. Since the batch order can affect the final result, we compute the average over 10 runs where the batches are randomly shuffled. Moreover, for each scenario, we provide the accuracy of the cumulative strategy (i.e., the current batch and the entire previous ones are used for training) as a target [4]. We do not use the term upper bound because in principle a smart sequential training approach could outperform a baseline cumulative training. In the following sections we report results only for object level classification task (the most difficult one), since both experiments lead to the analogous conclusions. Furthermore, in this study the models were trained on RGB data only (no depth information).

5.1 NI: New Instances

In this scenario the training batches coincides with the 8 sessions available in the training set. In fact, since each session includes a sequence (about 300 frames) for each of the 50 objects, training a model on the first session ad tuning it 7 times (on the remaining 7 sessions) is in line with NI scenario: all the classes are known since from the first batch and successive batches provide new instances of these classes to refine and consolidate knowledge. A baseline naïve approach for this scenario is simply continuing the SGD training as new batches become available. In Figure 6 we compare the baseline and cumulative approaches.

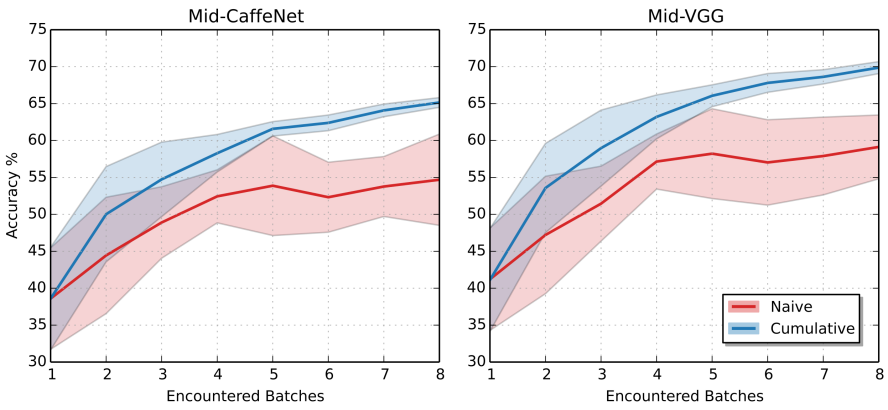

Figure 6: Accuracy results (averaged on 10 runs) for the naïve and cumulative strategies for Mid-CaffeNet and Mid-VGG. Colored areas represent the standard deviation of each curve. Tabular data available at `http://vlomonaco.github.io/core50`.

The accuracy gap between the naïve and the cumulative approach is here quite modest. In fact, with a careful tuning of the learning rate and number of iterations (early stopping), forgetting can be tamed in this scenario where the model memory is regularly refreshed with new poses (scale, view angle, occlusion, lighting, etc.). Similar findings have been reported for *NORB*, *COIL100* and *BigBrother* in the NI scenario [16] [15].

5.2 NC: New Classes

In this scenario, for each sequential batch, new objects (i.e. classes) to recognize are presented. Each batch contains the whole training sequences (8) of a small group of classes, and therefore no memory refresh is possible across batches. In the first batch we include 10 classes, while the remaining 8 batches contain 5 classes each. To slightly simplify the task, in each of the ten runs we randomly chose the classes with a biased policy which privileges maximal categorical representation (i.e.,

[4] To reduce computations, the number of runs for the NC and NIC cumulative strategy is reduced to 5 and 3 respectively.

spreading of objects of the same category in different batches). Defining an optimal testing protocol for NC scenario is not obvious. We initially considered three alternatives:

1. **NC – Partial Test Set**: at each evaluation step (i.e., after each training batch) the test set includes only patterns of the classes already presented to the network.
2. **NC – Full Test Set**: the test set is fixed and includes patterns of all the classes. Except for the last evaluation step, the model is (also) required to classify patterns of never seen classes.
3. **NC – Full Test Set with Rejection Option**: the test set is fixed and includes patterns of all classes, however the model has the possibility to reject a pattern if it believes the pattern does not belong to any of the known classes. Since the training set does not include "negative" examples we cannot add an extra neuron for the "unknown" class, and the rejection mechanism has to compare the max class probability with a given threshold.

Option 1. has the drawback that as we increase the number of classes in the test set the task becomes more complex and it is difficult to appreciate the learning trend and benefits of subsequent batches. Option 3. is the most realistic one for real applications but evaluation and comparison of different techniques is more difficult because at each step instead of a single point we have a ROC curve (accuracy also depends on the threshold). Considering that our aim is comparative evaluation among continual object recognition approaches we believe that option 2. is a good trade-off between simplicity and usefulness for the task. This option also maintains the test set coherent across all scenarios (NI, NC and NIC) so we decided to adopt it.

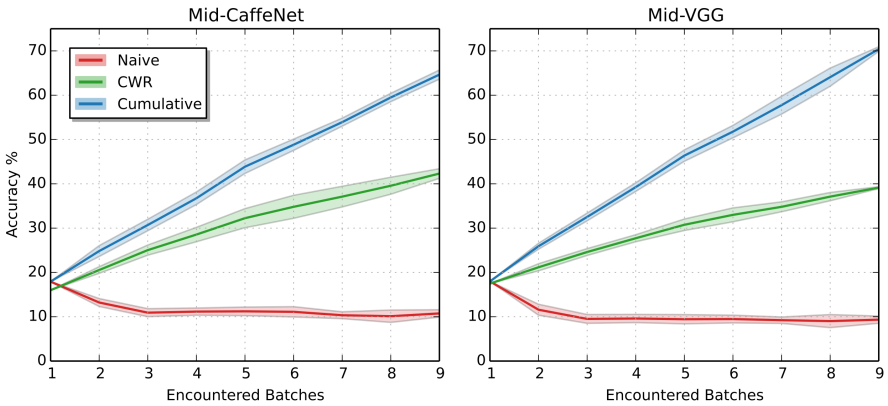

Figure 7: Mid-Caffe and Mid-VGG accuracy on NC scenario (average over 10 runs). Cumulative and Naïve approach are full depth models, while *CWR* lacks *fc6* and *fc7*. Colored areas represent the standard deviation of each curve. Tabular data available at `http://vlomonaco.github.io/core50`.

The naive approach in this scenario is not working at all. Graphs in Figure 7 show that the models completely forget the old tasks while learning the new classes: the initial accuracy drop is due to the larger size of the first batch w.r.t. the following ones. So we investigated other approaches and find out one (denoted as *CWR*: *CopyWeights with Re-init*) that, in spite of its simplicity, performs fairly well and can be used as baseline for further studies (see Figure 7).

In *CWR* we skip layers *fc6* and *fc7* and directly connect *pool5* to a final layer *fc8* (followed by *softmax*) while maintaining the weights up to *pool5* fixed. This allows isolating the subsets of weights that each class uses. During the training two sets of weights are maintained by the model for the *pool5* $\rightarrow$ *fc8* connections: **cw** are the consolidated weights used for inference and **tw** the temporary weights used for training; **cw** are initialized to 0 before the first batch, while **tw** are randomly re-initialized (default Gaussian distribution with std = 0.01, mean = 0) before each training batch. At the end of each batch training, the weights in **tw** corresponding to the classes in the current batch are copied in **cw**: this is trivial in this scenario because of the class segregation in different batches. In other words, **cw** can be seen as a sort of hippocampus where consolidated concepts are maintained, while **tw** as a short term working memory in the cortex used to learn new concepts without interfering with stable ones (see the interesting discussion in [13]).

Other simple approaches have been implemented and tested such as: *i*) *FW* where **cw** is not used and we just freeze in **tw** the class weights of the already encountered classes; *ii*) *CW* which is the same of *CWR* but without the re-init step; however as shown in Figure 8 the results obtained are significantly worse than *CWR*.

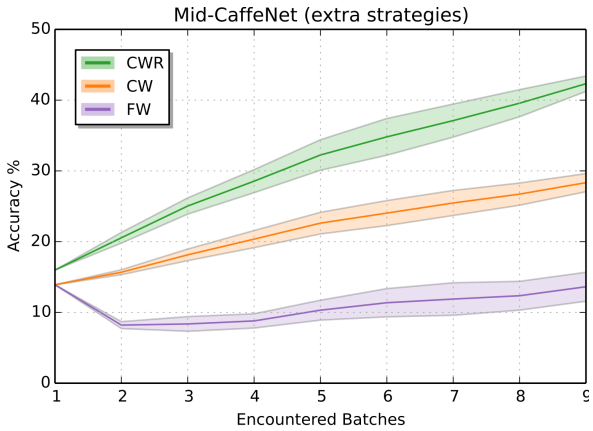

Figure 8: CWR compared with some variants: *FW* and *CW*. Colored areas represent the standard deviation of each curve.

5.3 NIC: New Instances and Classes

In the third and last scenario, both new classes and instances are presented in each training batch. This scenario is the closest to many real-world applications where an agent continuously learns new objects and refines the knowledge about previously discovered ones. As for NC scenario the first batch includes 10 classes, and the subsequent batches 5 classes each. However, only one training sequence per class is here included in a batch, thus resulting in a double partitioning scheme (i.e., classes and sequences). The total number of batches is 79. We maximized the categorical representation in the first batch but we left the composition and order of the 78 subsequent batches completely random. The test set is the same as in NI and NC scenarios.

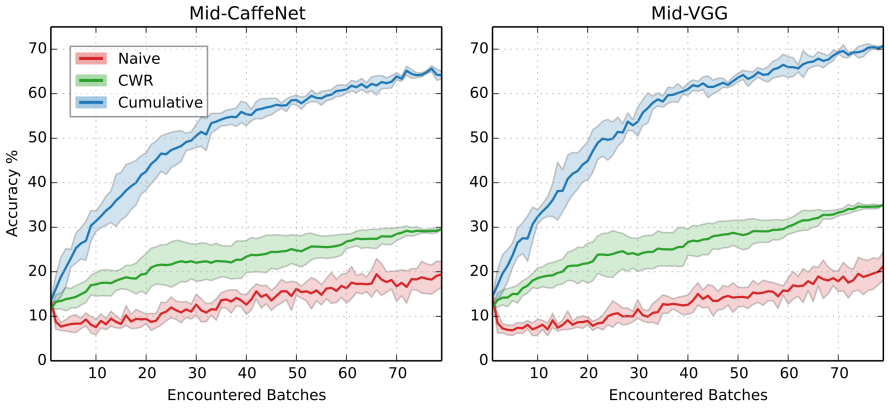

Figure 9: Mid-Caffe and Mid-VGG accuracy on NIC scenario (average over 10 runs). Colored areas represent the standard deviation of each curve. Tabular data available at `http://vlomonaco.github.io/core50`.

CWR approach for this scenario needs to be slightly adjusted. The first time a new class is encountered its **tw** weights are copied on **cw**, while at successive steps **cw** is updated as a weighted average. More

precisely, for each class i in the current batch:

$$cw[i] = \begin{cases} tw[i] & \text{if } updated[i] = 0 \\ \dfrac{cw[i] \cdot udpdates[i] + tw[i]}{updates[i] + 1} & \text{otherwise} \end{cases}$$

where $update[i]$ is the number of times class i has been encountered so far. Figure 9 reports the accuracy of the naïve, cumulative and *CWR* approaches. The graph clearly shows that this scenario is very difficult (*CWR* accuracy is about half of the Cumulative approach accuracy) and there is a big room for improvements.

6 Conclusion

Biological learning requires neither to store perceptual data streams nor to process them in a cumulative way; however, it effectively tackles incremental learning tasks where new object representations are continuously learned and consolidated and only the useless one are forgot. Artificial learning systems still lack these capabilities and new research is needed to fill the gap. In this paper we introduced a new dataset and associated benchmarks to support continuous learning studies in the context of object recognition.

As argued by many researchers our results also prove that naïve approaches such as incremental tuning cannot avoid catastrophic forgetting in complex real-world scenarios like NC and NIC. When testing new approaches on *CORe50*, the most important indicator is not the absolute accuracy on specific scenarios but the relative accuracy w.r.t. the corresponding cumulative approach. In fact, using a state-of-the-art full size CNN one can easily exceed absolute accuracy here reported for mid-size CNNs while only really effective continual learning techniques can significantly reduce the gap w.r.t. the corresponding cumulative approaches.

The proposed *CWR* approach (to be used as baseline for further studies) performs better than the naïve solution but the accuracy drops w.r.t. the cumulative approach is large and leaves much room for improvements. Directly connecting *pool5* to *fc8* and freezing the model parameters up to *conv5*, allowed us to easily disentangle the sets of weights which are relevant for the different classes, but more sophisticate techniques are needed in presence of multiple shared layers. *EWC* [11] and *LwF* [13] [28] are interesting approaches in this direction, and we intend to test them on *CORe50* in the near future. Other future steps in our plan are:

- Further extending the dataset in terms of classes and sessions;
- Combining motion and depth to provide better object segmentation;
- Classifying *CORe50* objects with models exploiting both RGB and depth information.

References

[1] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. YouTube-8M: A Large-Scale Video Classification Benchmark. *arXiv preprint arXiv:1609.08675*, 2016.

[2] Ali Borji, Saeed Izadi, and Laurent Itti. iLab-20M: A Large-Scale Controlled Object Dataset to Investigate Deep Learning. In *International Conference of Computer Vision and Pattern Recogniton (CVPR)*, pages 2221–2230, 2016.

[3] Raffaello Camoriano, Giulia Pasquale, Carlo Ciliberto, Lorenzo Natale, Lorenzo Rosasco, and Giorgio Metta. Incremental Robot Learning of New Objects with Fixed Update Time. *arXiv preprint arXiv:1605.05045*, 2016.

[4] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the Devil in the Details: Delving Deep into Convolutional Nets. *Proceedings of the British Machine Vision Conference (BMVC)*, pages 1–11, 2014.

[5] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, A Rusu, Alexander Pritzel, and Daan Wierstra. PathNet: Evolution Channels Gradient Descent in Super Neural Networks. *arXiv preprint arXiv:1701.08734*, 2017.

[6] Annalisa Franco, Dario Maio, and Davide Maltoni. The Big Brother Database : Evaluating Face Recognition in Smart Home Environments. In *Advances in Biometrics: Third International Conference (ICB)*, pages 142–150, 2009.

[7] Tommaso Furlanello, Jiaping Zhao, Andrew M. Saxe, Laurent Itti, and Bosco S. Tjan. Active Long Term Memory Networks. *arXiv preprint arXiv:1606.02355*, 2016.

[8] Jan-Mark Geusebroek, Gertjan J. Burghouts, and Arnold W.M. Smeulders. The Amsterdam Library of Object Images. *International Journal of Computer Vision*, 61(1):103–112, 2005.

[9] Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An Empirical Investigation of Catastrophic Forgeting in Gradient-Based Neural Networks. *arXiv preprint arXiv:1312.6211*, 2013.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, 2015.

[11] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. In *Proceedings of the National Academy of Sciences*, volume 114, pages 3521–3526, 2017.

[12] Yann. LeCun, Fu Jie Huang, and Leon. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 97–104, 2004.

[13] Zhizhong Li and Derek Hoiem. Learning without forgetting. In *Computer Vision - ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, volume 9908 LNCS, pages 614–629, 2016.

[14] Min Lin, Qiang Chen, and Shuicheng Yan. Network In Network. *arXiv preprint arXiv:1312.4400*, 2014.

[15] Vincenzo Lomonaco and Davide Maltoni. Comparing Incremental Learning Strategies for Convolutional Neural Networks. In *Artificial Neural Networks in Pattern Recognition: 7th IAPR TC3 Workshop (ANNPR 2016)*, pages 175–184, 2016.

[16] Davide Maltoni and Vincenzo Lomonaco. Semi-supervised Tuning from Temporal Coherence. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2509–2514, 2016.

[17] Davide Maltoni and Vincenzo Lomonaco. Semi-supervised Tuning from Temporal Coherence. *arXiv preprint arXiv:1511.03163*, 2016.

[18] Sameer A. Nene, Shree K. Nayar, and Hiroshi Murase. Columbia Object Image Library (COIL-100). *Technical Report*, 1996.

[19] Giulia Pasquale, Carlo Ciliberto, Francesca Odone, Lorenzo Rosasco, and Lorenzo Natale. Teaching iCub to recognize objects using deep Convolutional Neural Networks. In *Proceedings of Workshop on Machine Learning for Interactive Systems*, pages 21–25, 2015.

[20] Giulia Pasquale, Carlo Ciliberto, Lorenzo Rosasco, and Lorenzo Natale. Object Identification from Few Examples by Improving the Invariance of a Deep Convolutional Neural Network. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4904–4911, 2016.

[21] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In *CVPR 2016*, pages 779–788, 2016.

[22] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1137–1149, 2017.

[23] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive Neural Networks. *arXiv preprint arXiv:1606.04671*, 2016.

[24] Max Schwarz, Hannes Schulz, and Sven Behnke. RGB-D Object Recognition and Pose Estimation based on Pre-trained Convolutional Neural Network Features. *IEEE International Conference on Robotics and Automation (ICRA'15)*, (May):1329–1335, 2015.

[25] Arjun Singh, James Sha, Karthik S. Narayan, Tudor Achim, and Pieter Abbeel. BigBIRD: A Large-Scale 3D Database of Object Instances. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 509–516, 2014.

[26] Rupesh Kumar Srivastava, Jonathan Masci, Sohrob Kazerounian, Faustino Gomez, and Jürgen Schmidhuber. Compete to Compute. *Advances in neural information processing systems (NIPS)*, pages 2310–2318, 2013.

[27] Jeremy Steward, Derek Lichti, Jacky Chow, Reed Ferber, Sean Osis, and Canada Key. Performance Assessment and Calibration of the Kinect 2.0 Time-of-Flight Range Camera for Use in Motion Capture Applications. In *FIG Working Week 2015*, pages 1–14, 2015.

[28] Amal Rannen Triki, Rahaf Aljundi, Mathew B. Blaschko, and Tinne Tuytelaars. Encoder Based Lifelong Learning. *arXiv preprint arXiv:1704.01920*, 2017.

[29] Yuanyuan Zhang, Dong Zhao, Jiande Sun, Guofeng Zou, and Wentao Li. Adaptive Convolutional Neural Network and Its Application in Face Recognition. *Neural Processing Letters*, 43(2):389–399, 2016.

A Using pre-trained CNN with different input size

In the recent years, the pervasiveness of deep neural networks and the complexity of training such architectures on datasets of remarkable size has led to the proliferation of pre-trained models which represent a very good starting point for many customized solutions. However, this approach requires adapting problem-specific data to a fixed size architecture which was designed and optimized to solve another task. In the context of computer vision and object recognition, for example, it is very common to stretch images of arbitrary sizes to 227×227 pixels which is the typical input of well-known CNN models pre-trained on *ImageNet*: this often leads to highly distort the original patterns and significantly increases inference time. A more elegant (and efficient) approach is adapting a pre-trained model to work with input patterns of different size. This is straightforward for convolution and pooling layers thanks to local (shared) connections, but is much more problematic for fully connected layers, whose number of weights depends on the input image size. In this case, two main strategies can be used:

1. Applying fixed size pooling (global or pyramidal) over the last convolution/pooling layer as proposed in [10] [22] [14]. However, finetuning of upper levels might be necessary if the input scale changes dramatically or the original model was not designed with a fixed-size pooling layer at all.
2. Reusing the pre-trained network up to the last convolution layer and retraining the fully connected layers from scratch on the new task and input size. A typical approach is also to train an external classifier (e.g., SVM) from pooled features just after the last convolutional layer.

Independently of the network adaption to a different input size, when the problem classes change, the final softmax layer needs to be replaced and re-trained from scratch.

Since in our experiments we used the classic CaffeNet and VGG models (which have not been trained in a multi-scale fashion) and we aimed at fast processing, we opted for the second strategy. Hence, we reshaped the input volume to $3 \times 128 \times 128$, halved [5] the number of units in the fully connected layers *fc6* and *fc7* (from 4096 to 2048) and re-trained them from scratch. This results in a relevant

[5] As other authors we noted that such reduction has no significant impact on accuracy.

Table 3: Accuracy differences between reduced size CNNs (Mid) and the corresponding full-size models on the 50 classes task. All the models have been pre-trained on *ILSVRC-2012*. SVM training and CNN fine-tuning were performed on *CORe50*.

		Accuracy (object level: 50 classes)	
	CNN + SVM (on top of ...)	**fc6**	**pool5**
1	CaffeNet	63,46%	63,14%
2	Mid-CaffeNet		52,84%
3	VGG	69,03%	70,91%
4	Mid-VGG		59,25%
	CNN + Finetuning	**Accuracy (object level: 50 classes)**	
5	CaffeNet	75,97%	
6	Mid-CaffeNet	65,98%	
7	VGG	77,39%	
8	Mid-VGG	69,08%	

speedup at inference time (3.4× for CaffeNet and 4.67× for VGG). The resulting mid-size models are now suitable to be tuned on *CORe50* native 128 × 128 frames.

Table 3 summaries our findings. For the full-size models, extracting features from *fc6* or *pool5* is nearly equivalent in terms of accuracy (compare columns *fc6* and *pool5* for raw 1 and 3 in the table). So the lack of a fully pre-trained *fc6* in the mid-size models is not critical. However, in the experiments with SVM (rows 1:4), the mid-size networks loose about 10% accuracy with respect their original version. A similar gap (just slightly smaller for VGG) can be observed when the networks are finetuned (rows 5:8). The reason of such accuracy drop is not totally clear to us. On the one hand, if we consider finetuning experiments (rows 5:8), *fc6* and *fc7* have been pre-trained on a higher number of patterns in the full-size networks, and therefore it is reasonable to expect higher accuracy; on the other hand, if we consider *pool5* + SVM experiments, both the network exploits the same pre-training and stretching our input patterns from 128 × 128 to 227 × 227 (in principle) does not add new information.

We did similar experiments on other datasets (e.g., *NORB*, *COIL100*, *BigBrother*, *iCub*) and obtained close results: it seems that the zoomed image, even if a blurred, allow a more detailed feature extraction to be performed by the network. This can be due to the spatial scale of the filters learned on *ILSVRC-2012* or by a richer hierarchical representation (more neurons and link between neurons cover the object region). We believe that more investigations are necessary to fully understand the reasons and to make available pre-trained mid-size networks (for patterns whose native size is close to 128 × 128) which are competitive with full-size ones.